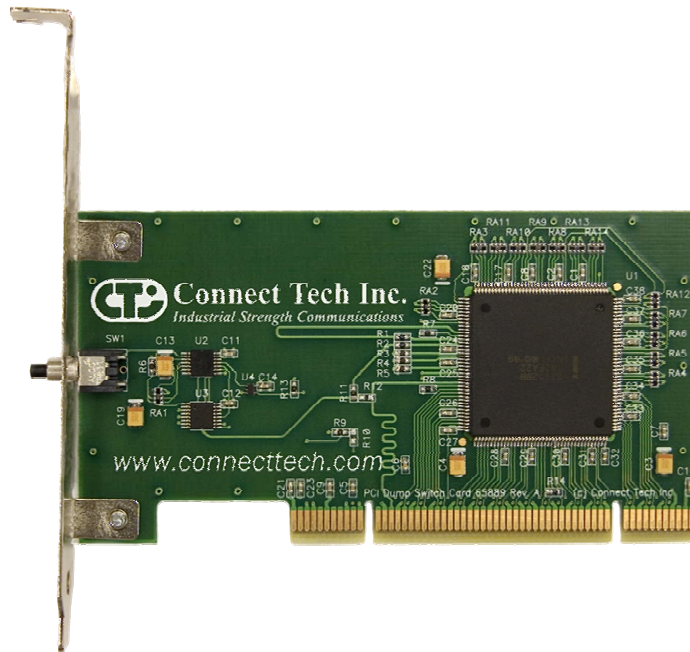




Connect Tech Inc.
Industrial Strength Communications

PCI Dump Switch Card



General Usage Notes

Connect Tech Inc
42 Arrow Road
Guelph, Ontario
N1K 1S6

Tel: 519-836-1291
Toll: 800-426-8979 (North America only)
Fax: 519-836-4878
Email: sales@connecttech.com
support@connecttech.com
URL: www.connecttech.com

Limited Lifetime Warranty

Connect Tech Inc. provides a Lifetime Warranty for all Connect Tech Inc. products. Should this product, in Connect Tech Inc.'s opinion, fail to be in good working order during the warranty period, Connect Tech Inc. will, at its option, repair or replace this product at no charge, provided that the product has not been subjected to abuse, misuse, accident, disaster or non Connect Tech Inc. authorized modification or repair.

You may obtain warranty service by delivering this product to an authorized Connect Tech Inc. business partner or to Connect Tech Inc. along with proof of purchase. Product returned to Connect Tech Inc. must be pre-authorized by Connect Tech Inc. with an RMA (Return Material Authorization) number marked on the outside of the package and sent prepaid, insured and packaged for safe shipment. Connect Tech Inc. will return this product by prepaid ground shipment service.

The Connect Tech Inc. Lifetime Warranty is defined as the serviceable life of the product. This is defined as the period during which all components are available. Should the product prove to be irreparable, Connect Tech Inc. reserves the right to substitute an equivalent product if available or to retract Life Time Warranty if no replacement is available.

The above warranty is the only warranty authorized by Connect Tech Inc. Under no circumstances will Connect Tech Inc. be liable in any way for any damages, including any lost profits, lost savings or other incidental or consequential damages arising out of the use of, or inability to use, such product.

Copyright Notice

The information contained in this document is subject to change without notice. Connect Tech Inc. shall not be liable for errors contained herein or for incidental consequential damages in connection with the furnishing, performance, or use of this material. This document contains proprietary information that is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced, or translated to another language without the prior written consent of Connect Tech, Inc.

Copyright © 2006 by Connect Tech, Inc.

Trademark Acknowledgment

Connect Tech, Inc. acknowledges all trademarks, registered trademarks and/or copyrights referred to in this document as the property of their respective owners.

Not listing all possible trademarks or copyright acknowledgments does not constitute a lack of acknowledgment to the rightful owners of the trademarks and copyrights mentioned in this document.

Table of Contents

Limited Lifetime Warranty	2
Copyright Notice.....	2
Trademark Acknowledgment.....	2
Table of Contents.....	3
Customer Support Overview.....	4
Contact Information	4
Introduction.....	5
Features	5
General Usage Notes.....	6
Enabling the NMI.....	6
Enabling the PCI Bridge SERR option.....	6
Using the NMI.....	6
QNX4 General Usage Notes	7
Enabling the NMI.....	7
Enabling the PCI Bridge SERR Option.....	7
Using the NMI in a QNX4 system	8
Debugger method.....	8
User action method	8
Windows NT, 2000, XP General Usage Notes	9
Setup.....	9
Troubleshooting.....	9
1. DP (disable-parity-check) bit is not reset in I/O location 0x61.....	9
2. NMI is masked by APIC or is not set properly.....	10
3. PCI bridges are not SERR enabled on command register at offset 0x04.....	10
4. NMI is not enabled on the NMI I/O location 0x70 (NMI Mask Register).....	11
Linux 2.6 General Usage Notes	12
Setup.....	12
Testing the PCI Dump Switch Card	13

Customer Support Overview

If you experience difficulties after reading the usage notes and/or using the product, contact the Connect Tech reseller from which you purchased the product. In most cases the reseller can help you with product installation and difficulties.

In the event that the reseller is unable to resolve your problem, our highly qualified support staff can assist you. Our support section is available 24 hours a day, 7 days a week on our website at: www.connecttech.com/sub/support/support.asp. See the contact information section below for more information on how to contact us directly. Our technical support is always free.

Contact Information

We offer three ways for you to contact us:

Mail/Courier

You may contact us by letter at:

Connect Tech Inc.
Technical Support
42 Arrow Road
Guelph, Ontario
Canada N1K 1S6

Email/Internet

You may contact us through the Internet. Our email and URL addresses on the Internet are:

sales@connecttech.com
support@connecttech.com
www.connecttech.com

Note:

Please go to the [Download Zone](#) or the [Knowledge Database](#) in the [Support Center](#) on the Connect Tech website for product manuals, installation guides, device driver software and technical tips.

Submit your technical support questions to our customer support engineers via the [Support Center](#) on the Connect Tech website.

Telephone/Facsimile

Technical Support representatives are ready to answer your call Monday through Friday, from 8:30 a.m. to 5:00 p.m. Eastern Standard Time. Our numbers for calls are:

Telephone: 800-426-8979 (North America only)
Telephone: 519-836-1291 (Live assistance available 8:30 a.m. to 5:00 p.m. EST, Monday to Friday)
Facsimile: 519-836-4878 (on-line 24 hours)

Introduction

The Universal PCI Dump Switch card is designed to enable developers to debug a crash situation even when a system freeze dictates a hardware reset. It increases developer productivity by eliminating the guess work normally required when debugging the cause of a system hang, by preserving valuable system information normally lost during reset. It includes an exterior push button to trigger a Non-Maskable Interrupt (NMI), which will enable a crash dump in Windows, or drop you into your operating system's debugger.

With this card, a software debugger can troubleshoot events even when the bug causes a complete lock of the operating system. (The PCI Dump Card does not reset the host computer.)

Features

- Universal 32-bit PCI card (PCI 2.3 compliant)
- Operating temperature range 0°C to 65°C
- Push button generation of a Non-Maskable Interrupt
- Transparent PCI to PCI bridge
- Standard MD1 format

General Usage Notes

Most operating systems will require the following to operate the PCI Dump Switch Card:

- Enable NMI support using a few IO instructions
- Set the SERR Enable bit on any PCI bridge chips between the PCI Dump Switch card and processor

The following instructions are intended as guidelines only. Every Operating System manages the implementation of NMI support in a different manner, so your setup may vary.

Enabling the NMI

Generally a few IO registers need to be set to allow the NMI to pass through the hardware to the processor. In a typical system, the following IO instructions will enable the NMI:

```
outp( 0x61, (inp( 0x61 ) & 0x0f) | 0x04 );
outp( 0x61, (inp( 0x61 ) & 0x0f) & ~0x04 );
outp( 0x70, 0x8f );
inp( 0x71, ); // dummy read
outp( 0x70, 0x0f );
inp( 0x71, ); // dummy read
```

Enabling the PCI Bridge SERR option

Some systems will need to have a register enabled on the PCI bridge chips located in the chain to the PCI Dump Switch card. This option is in the PCI command register and is called SERR. Also the ISA bus bridge (if present) and bus 0 PCI bridge may need to have SERR enabled. This will enable the SERR signal from the PCI card to be passed onto the main CPU.

In a typical system, the following will turn on the SERR Enable bit:

- Read in the Command register value for a given PCI bridge device
- OR this value with 0x100 to turn on the SERR Enable bit
- Store this value back to the Command register

To ensure the SERR PCI status is passed back to the CPU

- Look for the PCI Dump Switch bridge chip and find its connected bus
- Find the PCI bridge chip that supplied that bus and perform the same enable SERR bit change described above
- Repeat this sequence until you reach bus 0.
- If present, set the SERR bit for the PCI to ISA bridge chip.

Some computers do not require alteration of this SERR Enable bit (also known as SEE), but it is best if implementing a general solution that the PCI buses are scanned and have the bit enabled.

Using the NMI

The PCI Dump Switch card is typically used one of two ways:

- Use the native system debugger and have it pop up when the button on the PCI Dump Switch card is pushed
- Hook into the NMI interrupt in a program and have it take specific action, such as output data to a serial or parallel port, or save the status to some form of non-volatile memory

QNX4 General Usage Notes

These instructions are intended as guidelines only. QNX4 systems rely on the computer's BIOS to manage the implementation of NMI support and PCI resource, so your setup may vary. Contact support@connecttech.com if you require assistance.

Sample source code (`nmi_example.c`) is located on the CD that shipped with your AD005 and demonstrates how to perform the necessary initialization.

Most QNX4 systems will require the following two steps, outlined below:

- Enable NMI support
- Set the SERR Enable bit on any PCI bridge chips between the PCI Dump Switch card and processor

Enabling the NMI

Execute the following IO instructions to enable the NMI:

```

outp( 0x61, (inp( 0x61 ) & 0x0f) | 0x04 );
outp( 0x61, (inp( 0x61 ) & 0x0f) & ~0x04 );
outp( 0x70, 0x8f );
inp( 0x71, ); // dummy read
outp( 0x70, 0x0f );
inp( 0x71, ); // dummy read

```

Enabling the PCI Bridge SERR Option

The SERR option in the command registers of the PCI bridge chips between the PCI Dump Switch card and the ISA bus (if present) or bus 0 PCI bridge may need to be altered. This will enable the SERR signal from the PCI card to be passed onto the main CPU.

In a typical system, the following will turn on the SERR Enable bit:

- Read in the Command register value for a given PCI bridge device
- OR this value with 0x100 to turn on the SERR Enable bit
- Store this value back to the Command register

To ensure the SERR PCI status is passed back to the CPU, look for the PCI Dump Switch bridge chip and find the connected bus. Next, find the PCI bridge chip that supplied that bus and perform the same enable SERR bit change described above.

Repeat this sequence until you reach bus 0. This bit also needs to be set for the PCI to ISA bridge chip on some computers since the NMI is often implemented on the ISA bus.

Some computers do not require alteration of this SERR Enable bit (also known as SEE), but it is best if implementing a general solution that the PCI buses are scanned and have the bit enabled.

Using the NMI in a QNX4 system

There are two suggested ways of using the PCI Dump Switch board:

- Build a system debugger into the OS image and have it pop up when the button on the PCI Dump Switch card is pushed
- Hook into the NMI interrupt in a program and have it take specific action

Debugger method

To have the system debugger pop up, add the following to your QNX4 operating system build file then rebuild the image:

```
Sys/Debugger32  
$ Debugger32
```

When you boot the image created above, you may have to press `g<enter>` a few times to completely load QNX4.

User action method

The sample codes `main.c` and `example.c` provided on your AD005 CD provide examples of how the NMI can be set to trigger other actions, such as:

- store some parameters into non volatile memory
- output information directly to a device like a serial or parallel port

Windows NT, 2000, XP General Usage Notes

Setup

Ensure the system is set to enable system crash dumps via the Advanced Settings tab of Control Panel>System>Startup and Recovery Settings.

Set the system to generate a crash dump when requested through the PCI Dump Switch card. To do so, create the following registry value:

KLM\System\CurrentControlSet\Control\CrashControl\NMI\CrashDump. Set the DWORD entry value to 1 to enable the card and 0 to disable (default).

When the dump switch button is pressed, it should bring the user to the Windows debugger (WinDbg) if connected, otherwise it will produce a crash dump. If the debugger is running, you can choose to use WinDbg to debug the system immediately or begin a crash dump. If pressing the dump switch button does not result in being dropped into WinDbg or a crash dump, see the Troubleshooting section below.

Follow this link <http://www.microsoft.com/whdc/system/CEC/dmpsw.mspx> for more information on dump switch support for Microsoft Windows.

Troubleshooting

In most cases, your card will operate if you follow the instructions outlined on the previous page. If the card does not work, connect your system to WinDbg to examine and set the registers and I/O locations to the proper values, which may vary depending on the motherboard.

The following are some of the more common setup issues with Windows. Please contact support@connecttech.com for assistance if your issues persist after trying the following fixes.

1. DP (disable-parity-check) bit is not reset in I/O location 0x61 (NMI Status Register)
2. NMI is masked by APIC or is not set properly
3. PCI bridges are not SERR enabled on command register at offset 0x04
4. NMI is not enabled on the NMI enable/disable I/O location 0x70 (NMI Mask Register)

Note: Some motherboard/OS combinations may result in a delay of a few minutes from when the dump switch is engaged to the actual crash. This is due to a timer service that checks the validity of the NMI when the application runs during testing.

1. DP (disable-parity-check) bit is not reset in I/O location 0x61

I/O location 0x61 is known as the NMI status register. If bit 2 is set, the PCI Dump Switch card will not operate. Type `!b 61` in WinDbg to see the value. If bit 2 is set, reset it using the WinDbg command `ob`. Do not alter any other bit fields.

2. NMI is masked by APIC or is not set properly

Newer motherboards ship with APIC. Type `!apic` in WinDbg to see whether the NMI is masked. The following is a sample output from WinDbg:

```
kd> !apic

Apic @ ffe0000 ID:0 (50014) LogDesc:01000000 DestFmt:ffffff TPR D1
TimeCnt: 05f14d20clk SpurVec:1f FaultVec:e3 error:0
Ipi Cmd: 00040041 Vec:41 FixedDel Dest=Self edg high
Timer...: 000300fd Vec:FD FixedDel Dest=Self edg high masked
Linti0.: 0001001f Vec:1F FixedDel Dest=Self edg high masked
Linti1.: 000184ff Vec:FF NMI Dest=Self lvl high masked
TMR: 63, 83, B1, B4
IRR: 41, 63, B1, D1
ISR: D1
```

In this example, Linti1 NMI is masked. The status location is at offset `0xfe00360` and the status value is `0x000184ff`. In this case, bit 16 is set and therefore it is shown as masked.

Unmask the NMI in the APIC by `!ed [uc] fee00360 000084ff`. To check, use the `!apic` command. The following output should appear:

```
kd> !ed [uc] fee00360 000084ff
kd> !apic

Apic @ ffe0000 ID:0 (50014) LogDesc:01000000 DestFmt:ffffff TPR D1
TimeCnt: 05f14d20clk SpurVec:1f FaultVec:e3 error:0
Ipi Cmd: 00040041 Vec:41 FixedDel Dest=Self edg high
Timer...: 000300fd Vec:FD FixedDel Dest=Self edg high masked
Linti0.: 0001001f Vec:1F FixedDel Dest=Self edg high masked
Linti1.: 000084ff Vec:FF NMI Dest=Self lvl high
TMR: 63, 83, B1, B4
IRR: 41, 63, B1, D1, E3
ISR: D1
```

3. PCI bridges are not SERR enabled on command register at offset 0x04

Sometimes SERR are not enabled on PCI bridges on the bus route of the PCI Dump Switch card. Use the `!pci` command to see the values set for the command register.

```
kd> !pci 2 f

PCI Bus 0
00:0 8086:2530.04 Cmd[0106:.mb..s] Sts[2090:c....] Intel Host Bridge SubID:1028:010d
01:0 8086:2532.04 Cmd[0107:imb..s] Sts[00a0:.6...] Intel PCI-PCI Bridge 0->0x1-0x1
1e:0 8086:244e.04 Cmd[0107:imb..s] Sts[0080:.....] Intel PCI-PCI Bridge 0->0x2-0x3
1f:0 8086:2440.04 Cmd[000f:imb...] Sts[0280:.....] Intel ISA Bridge
1f:1 8086:244b.04 Cmd[0005:i.b...] Sts[0280:.....] Intel IDE Controller SubID:1028:010d
1f:2 8086:2442.04 Cmd[0005:i.b...] Sts[0280:.....] Intel USB Controller SubID:1028:010d
1f:3 8086:2443.04 Cmd[0001:i.....] Sts[0280:.....] Intel SMBus Controller SubID:1028:010d
1f:4 8086:2444.04 Cmd[0005:i.b...] Sts[0280:.....] Intel USB Controller SubID:1028:010d
PCI Bus 0x1
00:0 1002:5446.00 Cmd[0087:imb..] Sts[02b0:c6...] ATI VGA Compatible Controller SubID:1002:0409
PCI Bus 0x2
0a:0 8086:b152.00 Cmd[0107:imb..s] Sts[0290:c....] Intel PCI-PCI Bridge 0x2->0x3-0x3
0c:0 10b7:9200.78 Cmd[0117:imb..s] Sts[0210:c....] 3Com Ethernet Controller SubID:1028:010d
1f
```

The above shows the output for bus 0 to 2. The `Cmd` column is the value set for any PCI device in the command register. If bit 8 is set then SERR is enabled. In this case, all the bridges have

SERR enabled with the exception of the ISA bridge. You can view the details of the configuration registers value:

```
kd> !pci 100 0
```

```
PCI Bus 0
00:0 8086:2530.04 Cmd[0106:.mb..s] Sts[2090:c....] Intel Host Bridge SubID:1028:010d
Config Space: (Bus: 0 Device: 0 Func: 0)
00: VendorID 8086 Intel Corporation
02: DeviceID 2530
04: Command 0106 MemSpaceEnable BusInitiate SERREnable
```

The bridge is SERR enabled as shown:

```
04: Command 0106 MemSpaceEnable BusInitiate SERREnable
```

All the bridges should be SERR enabled. In this case the ISA bridge is not SERR enabled:

```
1f:0 8086:2440.04 Cmd[000f:imb...] Sts[0280:.....] Intel ISA Bridge
```

If the NMI is on the route of this ISA bridge, then the card will not produce an NMI interrupt. To SERR enable the ISA bridge, use the command `!ecw 0.1f.0 4 10f`.

Check the value again. In the following example, the ISA bridge is now SERR enabled.

```
kd> !ecw 0.1f.0 4 10f
```

```
kd> !pci 2 f
```

```
PCI Bus 0
00:0 8086:2530.04 Cmd[0106:.mb..s] Sts[2090:c....] Intel Host Bridge SubID:1028:010d
01:0 8086:2532.04 Cmd[0107:imb..s] Sts[00a0:.6...] Intel PCI-PCI Bridge 0->0x1-0x1
1e:0 8086:244e.04 Cmd[0107:imb..s] Sts[0080:.....] Intel PCI-PCI Bridge 0->0x2-0x3
1f:0 8086:2440.04 Cmd[010f:imb..s] Sts[0280:.....] Intel ISA Bridge
1f:1 8086:244b.04 Cmd[0005:i.b...] Sts[0280:.....] Intel IDE Controller SubID:1028:010d
1f:2 8086:2442.04 Cmd[0005:i.b...] Sts[0280:.....] Intel USB Controller SubID:1028:010d
1f:3 8086:2443.04 Cmd[0001:i.....] Sts[0280:.....] Intel SMBus Controller SubID:1028:010d
1f:4 8086:2444.04 Cmd[0005:i.b...] Sts[0280:.....] Intel USB Controller SubID:1028:010d
PCI Bus 0x1
00:0 1002:5446.00 Cmd[0087:imb...] Sts[02b0:c6...] ATI VGA Compatible Controller SubID:1002:0409
PCI Bus 0x2
0a:0 8086:b152.00 Cmd[0107:imb..s] Sts[0290:c....] Intel PCI-PCI Bridge 0x2->0x3-0x3
0c:0 10b7:9200.78 Cmd[0117:imb..s] Sts[0210:c....] 3Com Ethernet Controller SubID:1028:010d
1f
```

4. NMI is not enabled on the NMI I/O location 0x70 (NMI Mask Register)

If, after performing the first three steps, the card does not produce a crash dump, then the NMI may not be enabled at all at offset location 0x70. This location is a write-only location. Bit 7 is responsible for enabling NMI. Use `0b 70 0f` and then a dummy read on offset 0x71 by `ib 71`.

Linux 2.6 General Usage Notes

Setup

The Linux usage notes assume a Linux 2.6 kernel with kernel debugger (KDB) installed, complete with kernel source code. Please note that some steps may differ based on individual configurations.

To begin, ensure that PCI access is enabled (CONFIG_PCI=y and CONFIG_PCI_GOANY=y), and that the KDB is on (CONFIG_KDB=y and #CONFIG_KDB_OFF is not set). These options are located in your kernel configuration file.

Modify the file (source_tree)/arch/i386/kernel/traps.c, adding the following to the function do_nmi(...)

```
#ifdef CONFIG_KDB
    kdb_diemsg = "AD005 nmi";
    kdb(KDB_REASON_OOPS, 0, regs)
#endif
```

This code will start the kernel debugger when the dump switch button on the AD005 card is pressed. Rebuild the kernel and copy it to the appropriate location. Copy the system map file as the KDB needs to provide up-to-date information. Reboot the computer and select the modified kernel for the changes to take effect.

Press the button on the AD005 card. A KDB prompt should appear: kdb>

If the prompt doesn't appear, repeat the instructions above.

Note: Newer motherboards with APIC may need to clear bit 2 from the register **0x61** **outb(inb(0x61)&0xFB, 0x61)** and **outb(inb(0x70)&0xFB, 0x70)** to enable the card to trigger an interrupt.

If the prompt appears, type in the g command and press enter:

```
kdb>g
```

Ignore any warnings and enter the g command again.

```
kdb>g
```

This will return you to the Linux command prompt.

Testing the PCI Dump Switch Card

The sample below illustrates how you can use the PCI Dump Switch Card to debug your own software. The following program snippet simulates a situation where interrupts are disabled and a program is stuck in an infinite loop. Use this to test the dump switch card in a lock situation. (Note that running this program will lock your computer. Ensure your PCI Dump Switch card is configured to generate the kernel debugger before you begin.)

```
int main(void)
{
    iopl(3);
    /* set APIC and SERR stuff */
    outb(inb(0x61)&0xFB, 0x61);
    outb(inb(0x70)&0x7F, 0x70)
Label1:
    cli();
    goto Label1;
}
```

Save and compile the above: `Gcc -g <filename>.c -o <filename>`
The `-g` flag will add debugging information required later.

Run the above program from a local terminal. The keyboard should lock up.
Press the switch on the AD005. The `kdb>` prompt should appear after a second or two.

From the kernel debugger:
`kdb>bt` <enter>

This will present information and a stack trace of the current running process.

To see the registers of the processor, type:
`kdb>rd` <enter>

Record the value found in the `eip` register (0x080483BE).

For assembly language instructions (beginning at the address specified), type:
`kdb>id 0x080483BE` (the `eip` register value) <enter>

If you can read the assembly language, you can jump out of infinite loops using the `kdb go` command with an address of an instruction beyond the end of the loop. For example,
`kdb>g 0x080483FE` <enter>
will start the program at address 0x080483FE.

Once the problem address is located, return to the Linux command prompt. (This may require a reboot.)

Enter `#objdump -dS <executable file name> | less`

This will pipe your program to the `less` program, where you will see the address values. Finding the address 0x080483BE reveals your code in mixed assembler and C code. This will lead to where the bug occurred. You can now fix the bug.